Comparative Experimental Analysis of the Quality-of-Service and Energy-Efficiency of VMs and Containers' Consolidation for Cloud Applications

Ismael Cuadrado-Cordero, Anne-Cécile Orgerie, Jean-Marc Menaud

Comparative Experimental Analysis of the Quality-of-Service and Energy-Efficiency of VMs and Containers' Consolidation for Cloud Applications

Ismael Cuadrado-Cordero
IMT-A, INRIA, LS2N, Nantes, France
ismael.cuadrado-cordero@inria.fr

Anne-Cécile Orgerie
CNRS, IRISA, Rennes, France
anne-cecile.orgerie@irisa.fr

Jean-Marc Menaud
IMT-A, INRIA, LS2N, Nantes, France
menaud@mines-nantes.fr

***Abstract*—The consolidation of services is a widely accepted technique for IaaS Cloud providers to reducing energy consumption and improving the utilization of their resources. This technique is based on distributing all services in the minimum amount of servers. This way, the overall energy consumption of the datacenter is reduced, as less servers are needed to be active. Traditionally, research has focused on strategies for consolidation of Virtual Machines (VMs), but containers are changing the landscape of Cloud services. Containers are expected to optimize the consolidation of services by reducing the amount of needed resources, thus allocating more services using less servers. However, while multiple research works have been produced in the Energy Efficiency (EE) achieved through consolidation of VMs, there is yet no experimental work on how consolidation of containers affects EE, when assuming a given Quality-of-Service (QoS) to the user. In this paper we show an experimental analysis on the effects of consolidation of containers in the QoS and EE, compared to the consolidation of VMs. We demonstrate that the consolidation of containers is indeed more optimal than the one of VMs, both in terms of QoS and EE. Consecutively, we analyze how the degradation of the service is produced both in QoS and EE, and we show how QoS is the variable which is more affected by consolidation. This work provides the necessary scientific background on consolidation of two widely used virtualization technologies, and we believe it is useful for future works on the optimization of resources in datacenters.**

I. Introduction

Cloud Computing has become one of the main technologies in the Internet, particularly at an Infrastructure-as-a-Service (IaaS) level, due to its virtualization of resources. One of the main objectives of virtualization is that several clients can execute their services on the same physical machine (server), keeping these services isolated from each other. Virtualization techniques consequently advocate for consolidation that allows to gather several virtual environments on the same server to optimize resources. Currently, virtualization of resources is done mainly through two technologies: Virtual Machines (VMs) and containers. VMs emulate all the functionalities of a physical machine, while containers are instances running all on the host Operating System's kernel. Containers are a more lightweight virtualization technology than VMs, and have seen a growing popularity in the last years.

Recently, several research works have evaluated from different perspectives the performance of containers and VMs as virtualization technologies. All these works focus on the performance of a fixed number of services running on both technologies. The research challenge taken in this work is to experimentally compare how many services can run in the same server, respecting a given Quality-of-Service (QoS) and Energy Efficiency (EE), when using different virtualization technologies. We experimentally compare the performance of VMs and containers, and demonstrate that the use of containers is preferable both in terms of QoS and EE. Our evaluation is based on a quantitative analysis and comparison between real deployments of a typical Cloud application on VMs and containers. Finally, we show that, for a given application, it is possible to deploy a higher number of virtualized environments using containers than VMs, assuming a minimum acceptable QoS. This way the prospective energy savings of using containers are increased, as less servers are needed to run the same services. This paper enhances existing literature, by evaluating the performance of both technologies under different numbers of services, and establishing a relation between QoS and EE in a consolidated environment.

The remainder of the paper is as follows: Section II provides the main necessary definitions, and shows the most relevant related work. Section III describes the experimental setup used in this work. Section IV evaluates and explains the result of our experiments. Section V discusses the results obtained during experimentation. Finally, Section VI highlights the main findings and lessons learned from our work.

II. Background

A. Context and Motivation

On the one hand, the increase in demand of resources and the growing number of users have defined new challenges to current IaaS Cloud datacenters, such as QoS for the clients or scalability of their infrastructures [1]. On the other hand, datacenters have great demands for energy and are estimated to consume more than 2.4% of electricity worldwide with a global economic impact of $30 billion [2].

Cloud providers addressed the problems of energy consumption and scalability through virtualization and consolidation. This way, they consolidate services on several active servers, thus using less resources [3]. However, placing multiple virtualized services on the same server has been demonstrated to lead to a QoS degradation. While this limit has been studied for other virtualization techniques [4], with the appearance of containerization technologies it becomes necessary to evaluate this behavior in consolidated environments using containers, and how this consolidation affects QoS and EE.

B. Virtual Machines

A Virtual Machine was initially defined by Popek and Goldberg, as "an efficient, isolated duplicate of a real computer machine" [5]. VMs virtualize multiple guest Operating Systems on top of the same host OS. To manage the several guest OSs, a specific software is deployed in the host, called a hypervisor. A hypervisor may offer full virtualization, where all the underlying hardware to the VM is simulated; or hardware-assisted virtualization, where the host processor offers support for the virtualization techniques. Literature work assesses that hardware-assisted virtualization offers a better performance than full virtualization [6]. As described in Section II-D, KVM[1] is one of the most researched hypervisors.

C. Containers

Containers are a type of virtualization which aims at reducing the resources utilization done by VMs, especially in memory. In order to achieve that, containers are virtualized at the kernel level, where the OS's kernel manages multiple isolated user-space instances at the same time. In the last years, containers have won popularity as an alternative to VMs [7]. Containers are expected to reduce overhead and improve flexibility of virtualized environments, because services running on containers directly interact with the OS, without redirecting their instructions to the hypervisor. However, the containers' virtualization does not isolate services as well as VMs, which may pose security issues in some Cloud deployments. Of all existing implementations of containers, Docker[2] are one of the most researched in literature, as discussed in Section II-D.

D. Related work

Quality-of-Service of containers has been studied in literature. In 2014, Seo et al. [8] compared containers to VMs in terms of disk utilization, boot time and operation speed. To compare both disk utilization, authors used a service generating several files, and replicated this service using KVM and Dockers. Authors also measured boot time using similar virtual images in both technologies. Finally, for speed comparison, they calculated the mathematical operation 100000! using Python. Their results show that containers outperform VMs in all experiments. A year later, Felter et al. [9] showed that the utilization of containers has a lighter weight on CPU than using VMs. Their experiments involved running one MySQL service using different configurations on Docker, and comparing it to a KVM-managed VM. According to their results, while both technologies are light on the CPU usage, VMs utilized as much as 38% more CPU time than Docker.

Energy Efficiency of both, containers and VMs, has been also evaluated. In 2015, Morabito [10] performed several experiments using different dedicated benchmarks (CPU intensive; memory intensive; and network intensive) on the power consumption of VMs (KVM and Xen) and containers (Docker and LXC[3]). The authors evaluate eight simultaneous services in the same physical machine. Results of this work show that containers have a lighter power consumption than VMs. Comparing among the same technologies, it shows that KVM makes a more efficient use of energy than Xen, while dockers slightly outperforms LXC on the overall energy efficiency. These results are supported by the work of Jiang et al. [11]. In their work, the authors evaluate the energy consumption of different hypervisors, including KVM and Xen, in different hardware machines with a varying workload. They conclude that KVM consumes less energy than Xen in the evaluated architectures.

To the best of our knowledge, there is no work which relates energy consumption and performance of containers when simultaneous services are consolidated in the same server. Also, no work in literature offers an empirical limit of containers running on the same physical machine, as it has been already addressed for VMs [4]. We believe that an empirical investigation on the performance of containers is useful to better understand their potential.

III. Experimental setup and methodology

We focus on evaluating the impact that consolidating multiple virtualized services on the same server has on QoS and EE. The service to evaluate is a LAMP stack (Linux-Apache-MySQL-PHP), virtualized in the same host server using different technologies, as it is a very extended archetypal model of existing web services. LAMP is named after the four open-source components from which is formed: Linux (OS), Apache (HTTP Server), MySQL (database), and PHP (programming language). Our deployment runs MySQL Version 14.14 Distr. 5.5.54; Apache 2.0; and PHP Version 5.4.45. Each service runs the web service benchmark RUBiS[4]. RUBiS benchmark is accepted in literature as a good example of a typical Cloud application. Specifically, our deployment of this benchmark simulates multiple concurrent users in an online auction market. Each client machine runs one experiment against one service during a total 34 minutes and 12 seconds each time. Each experiment simulates an increment of users connecting to a single service (from 0 to 300 users) during 2 minutes; during 30 minutes host an fixed number of users (300 users); and reduce the number of users (from 300 to 0) in 2 minutes and 12 seconds. That is, if 10 services are running on the same host, the server is connecting up to 3 000 users

[1]Kernel-based Virtual Machine https://www.linux-kvm.org/
[2]https://www.docker.com
[3]Linux Containers - https://linuxcontainers.org/
[4]Rice University Bidding System. http://rubis.ow2.org

simultaneously during the duration of the experiment. Each user has a randomized access pattern to the web service. The possible actions of each user in this service are: access to the home page; browsing items, categories or regions; and view items. The database contains 10,000 items and 1,000,000 users. Each user has left up to 20 commentaries to different items and each item has a maximum of 20 bids from different users.

Our experiments were run using the Taurus cluster available in the French experimental testbed Grid'5000 [12]. This cluster is formed by several identical Dell PowerEdge R720, each one with 2 Intel Xeon E5-2630 6-Core CPUs (2.3GHz / 6256 KB (L2) + 15 MB (L3) / 7.2 GT/s QPI); 32 GB of memory and 600 GB of disk. The connections between nodes work on 10 Gigabit Ethernet interfaces, all directly connected to a Dell Force10 S4810 switch, as depicted in Figure 1. The cluster has been divided into one server and several clients. Both server (host OS) and clients run Debian GNU/Linux 8.7 (jessie) x86_64.

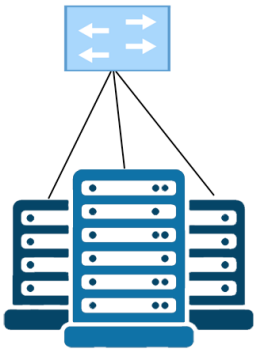

Fig. 1: Interconnection of servers in Taurus Cluster

Omegawatt wattmeters specially furnished for this cluster are used for the energy measurements. Every wattmeter provides one energy measurement every second, with an error margin of 0.125 Watts. The energy measurements do not include networking devices (i.e. router) because, as the network remains the same for all the experiments, the EE remains unchanged [13], [14].

Due to fairness in the comparison, our experimentation is based on the technologies which have been proven in literature to be most energy efficient both for VM and containers. So, the technologies to compare are:

VM: We used a hardware-assisted virtualization, using KVM as a hypervisor. Each LAMP stack is implemented in a different VM as defined above, each one holding its own copy of the database. VMs' instances run Debian GNU/Linux 7.11 (wheezy) x86_64, allocating 2 vCPUs and 2 GB of memory.

Containers: Containers have been implemented using Docker version 17.03.0-ce. Each Docker container incorporates the same LAMP stack and RUBiS configuration. Each container runs a version of Debian GNU/Linux 7.11 (wheezy).

Successive experiments deploy respectively 1, 2, 5, 10, 15, 20 and 25 different services in the same server. For each couple of experiments (VMs and containers), the same server has been used to avoid measurements' disruptions due to heterogeneity that may have appeared in the cluster's lifetime. Each experiment is launched three times to strengthen their statistical significance.

IV. Experimental Results

In our first set of experiments we address QoS of a consolidated set of services in the different technologies. We focus on the response time (latency) experienced by users, as this is the variable of the QoS which represents the "single greatest contributing factor to spatial and temporal inconsistencies experienced by end users in the virtual world" [15]. According to literature [16], [17], latency in web services can go up to 1 second before interrupting the user's train of thought. On the other hand, the standard in industry advocates for a maximum latency of 2 seconds [18]. Finally, Brutlag et al. [19] found that users of search engines tolerate up to 3 seconds of latency before changing technologies. To assess the consolidation of services under both technologies, we evaluate the evolution of QoS when an increasing number of virtualized services is deployed in a server. We take for latency reference values: 1 000 ms, 2 000 ms and 3 000 ms.

Figure 2 shows the values of the average latency experienced by the user, under a different number of services deployed on the same server, comparing KVM and Docker. As shown, the 1,000 ms threshold is already reached by KVM when more than 15 simultaneous services are run on the same server. Similarly, the 2,000 ms threshold is surpassed when deploying 17 VMs on the same server, while the 3,000 ms' one is surpassed at 19 VMs. Comparing this result with Docker we see that a higher number of services can be deployed on the same server before the thresholds are reached. In fact, the 1,000 ms threshold is surpassed when deploying 16 containers and the 2,000 ms one at 20 containers. Thus, the same server can host up to 3 more services in the form of containers while still providing an acceptable delay (second threshold). Also, the degradation of the response time using KVM increases more rapidly than using Docker. It is shown how the degradation of the QoS of KVM dramatically increases once reached 15 simultaneous VMs. On the other hand, up to 24 containers can be deployed before reaching the 3,000 ms threshold, compared to 19 VMs. This represents an increase of a 26% of consolidated services before surpassing the third threshold.

An excessive delay also affects the number of successful interactions between the users and the service. An interaction is a request from the user (i.e. new page) which has been responded by the server. A user waits for a request to be answered before sending a new one. As each client runs the experiment for a fixed amount of time, the higher the latency the less requests can be sent. Figure 3 shows the evolution on requests managed by the server when the number of services running on it increases. Once again, it is observed a degradation in KVM after deploying more than 15 services.

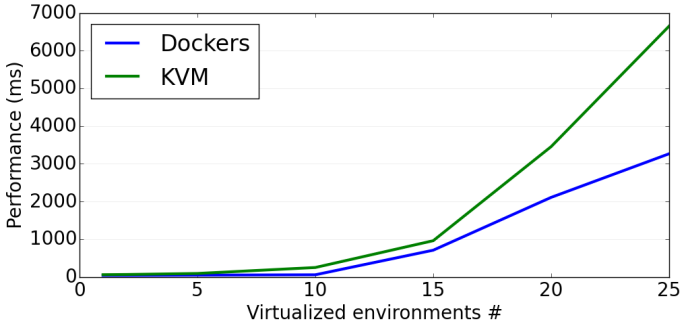

Fig. 2: Evolution of latency over an increasing number of virtualized services

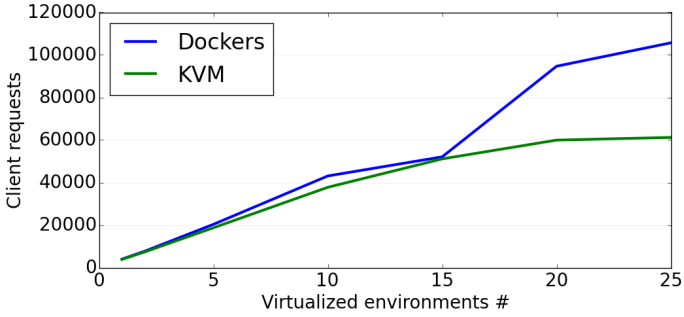

Fig. 3: Evolution of interactions managed by the server over an increasing number of services

Finally, Figure 4 shows our evaluation of energy consumption evolution under an increasing number of consolidated services, both using KVM and Docker. In these experiments it is observed that Docker makes a better utilization of energy resources. As shown, energy consumption significantly increases, using either KVM or Docker, for the range between 1 and 5 services. It is also noticed that, in this range, Docker always consumes less energy than KVM. In the range between 5 and 10 services, both technologies stabilize their consumption. Finally, in the range from 10 to 25 services, the energy consumption of KVM exponentially increases, while the consumption of Docker remains stable (slightly increases).

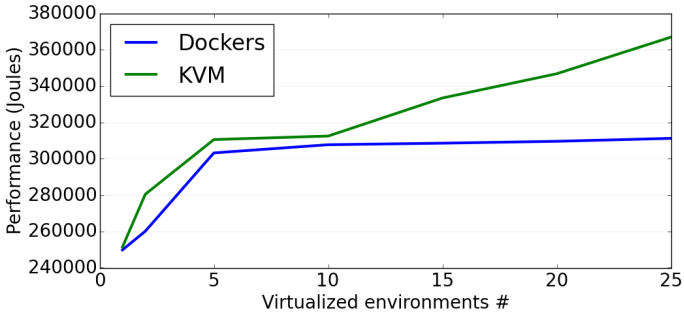

Fig. 4: Evolution of energy consumption over an increasing number of services

These results are explained by the different utilization of resources done by both technologies. To demonstrate this, we evaluate the utilization of resources done by both technologies, running an extra set of experiments focused on resources utilization. During the same experiment we sequentially run experiments deploying 5, 10 and 15 simultaneous services on the same server for both Docker and KVM. Thus, every experiment runs for 1 hour and 50 minutes. Data has been retrieved using Ganglia Version 3.1.7[5].

First, we focus on why KVM consumes more energy than Docker, in the range between 1 and 10 services. As shown above, before its energy consumption booms, KVM consumes more energy than Docker. This expected, and is explained by the use of resources, especially CPU and memory, which is made by the different technologies. Figure 5 depicts the consumption of the CPU idle along the experiment. As shown, KVM makes a more demanding CPU utilization, which is later translated into higher energy consumption. This cost reflects the cost of the hypervisor, not existing for containers.

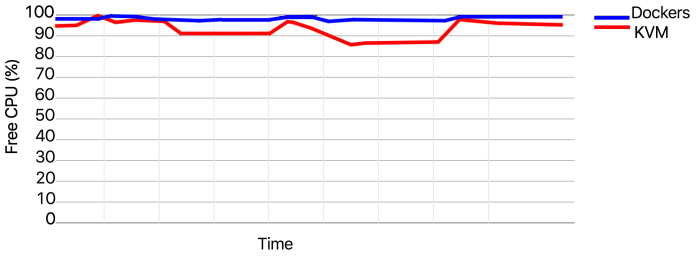

Fig. 5: Evolution of CPU idle when deploying several services

Second, we investigate why both EE and QoS degrades faster when hosting more than 10 VMs per host. As shown in Figure 6, the utilization that KVM makes of the memory resources is significantly higher than the utilization made by Docker. As KVM runs out of free memory, it starts allocating swap memory in disk, which affects QoS and increases energy consumption. This is explained by the difference in how both technologies manage the memory. KVM allocates a fixed amount of memory for each VM, which is used for the guest OSs. Furthermore, the virtualization of this memory is not as optimal as it is when managed by the host OS. On the other hand, Docker treats memory from the host OS's kernel, not allocating fixed virtualized blocks of memory per instance, so it is better managed.

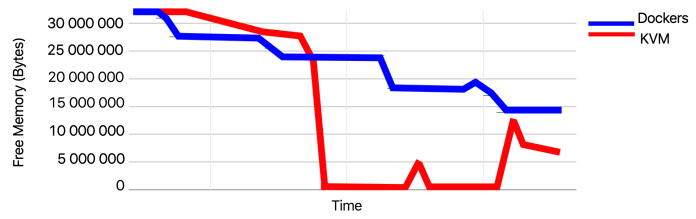

Fig. 6: Evolution of free memory when deploying several services

[5]Ganglia - http://ganglia.sourceforge.net

V. Discussion

To better compare both technologies, we point out different relations between QoS and energy consumption. First, we define the relation between clients' requests and energy consumption (interactions per joule) according to the number of consolidated services, as shown in Equation 1.

$$IntEnergy = \frac{numberofinteractions}{energy\ consumption} \quad (1)$$

As shown in Table I, for up to 10 services, the interactions per joule remain similar between both technologies. After 10 services we observe that both ratios start to differ, getting their biggest gap at 20 services. This value is consistent with the results shown in the previous Section. This is expected, and is explained because for 20 services, Docker offers a latency much smaller than KVM. We also notice how, for the case of KVM, the optimal number of services is found around 15 (as after the ratio growing slows down). On the other hand, the optimal behavior seems to be found in 20 containers, before the ratio growing slows down. As we deploy more services in the same server it manages more requests, but also increases its energy consumption. Thus, as shown, the winning in interactions per joule obtained by consolidating more services decreases and points at a plateau.

	1	2	5	10	15	20	25
KVM	0.015	0.026	0.06	0.12	0.15	0.17	0.189
Docker	0.0159	0.029	0.067	0.14	0.169	0.3	0.34

TABLE I: Relation between interactions and energy consumption (int/joule)

Second, we evaluate the degradation ratio between latency and energy consumption. This value relates the degradation in the average response time with this of the energy used during the benchmark; and is measured in milliseconds per joule. We calculate this relation through Equation 2. When deploying a number X of services, we compare deg_ratio_X to deg_ratio_{X-1}. An increase in the value of this variable respect to the former value shows that the latency degrades faster than the energy consumption increases, while a decrease implies the opposite. We use this relation to evaluate which variable (either QoS or EE) is more affected by the consolidation of services.

$$Deg\_ratio = \frac{1000 \times latency}{energyconsumption} \quad (2)$$

As a basic degradation ratio (one service), KVM provides 0.23 ms/joule. On the other hand, the performance of Docker is of 0.15 ms/joule. As the number of simultaneous services increases, so does the degradation, especially the degradation of latency, as shown in Table II. The experiments on KVM show a higher degradation in QoS than it does in EE, especially after reaching 10 simultaneous services. On the other hand, after reaching 10 simultaneous services, Docker degrades mostly its QoS, but more lightly than KVM.

	1	2	5	10	15	20	25
KVM	0.23	0.224	0.29	0.793	2.876	9.96	18.118
Docker	0.15	0.148	0.152	0.178	2.29	6.808	10.488

TABLE II: Relation between QoS and energy consumption degradation (ms/joule)

VI. Conclusions and future work

Since their introduction, containers have gained popularity as a lightweight virtualization technology. Containers promise a better QoS than other virtualization technologies, such as Virtual Machines. Furthermore, given that they require less resources than VMs, it is expected that more services can be consolidated in the same server, which reduces energy consumption, as less servers are needed to run the same amount of services.

In this work we compared the performance of VMs and containers when consolidating multiple services, in terms of QoS and EE. Our experiments compared two broadly recognized virtualization technologies: KVM for the VM approach, and Docker for the containers. We conclude that Docker outperforms KVM both in QoS and EE. According to our measurements, Docker allows running up to a 21% more services than KVM, when setting a maximum latency of 3,000 ms. In this configuration, Docker offers this service while using a 11.33% less energy than KVM. At a datacenter level, the same computation could run using less servers and less energy per server, accounting for a total of a 28% energy savings inside the datacenter.

As further work, we would like to evaluate the costs of migrating containers. Thus, we plan on extending our experimentations to include the casuistry where containers are migrated to dynamically consolidate services. We would like to compare the costs of these migrations and relate them with the utilization of the servers.

Acknowledgments

Experiments presented in this paper were carried out using the Grid'5000 experimental test-bed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

References

[1] K. Bilal, S. U. R. Malik, S. U. Khan, and A. Y. Zomaya, "Trends and challenges in cloud datacenters," *IEEE Cloud Computing*, vol. 1, no. 1, pp. 10–20, 2014.

[2] G. I. Meijer, "Cooling energy-hungry data centers," *Science*, vol. 328, no. 5976, pp. 318–319, 2010.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[4] M. Callau-Zori, L. Samoila, A.-C. Orgerie, and G. Pierre, "An experiment-driven energy consumption model for virtual machine management systems," IRISA ; Université de Rennes 1 ; CNRS, Research Report RR-8844, Jan. 2016.

[5] J. E. Smith and R. Nair, "The architecture of virtual machines," *Computer*, vol. 38, no. 5, pp. 32–38, May 2005.

[6] L. Nussbaum, F. Anhalt, O. Mornard, and J.-P. Gelas, "Linux-based virtualization for HPC clusters," in *Montreal Linux Symposium*, Montreal, Canada, Jul. 2009.

[7] Docker, "Docker community passes two billion pulls!" 2016. [Online]. Available: https://blog.docker.com/2016/02/docker-hub-two-billion-pulls

[8] K.-T. Seo, H.-S. Hwang, I.-Y. Moon, O.-Y. Kwon, and B.-J. Kim, "Performance comparison analysis of linux container and virtual machine for building cloud," *Advanced Science and Technology Letters*, vol. 66, no. 105-111, p. 2, 2014.

[9] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and linux containers," in *2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, March 2015, pp. 171–172.

[10] R. Morabito, "Power consumption of virtualization technologies: An empirical investigation," in *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, Dec 2015, pp. 522–527.

[11] C. Jiang, D. Ou, Y. Wang, X. You, J. Zhang, J. Wan, B. Luo, and W. Shi, "Energy efficiency comparison of hypervisors," in *2016 Seventh International Green and Sustainable Computing Conference (IGSC)*, Nov 2016, pp. 1–8.

[12] Grid5000, "Lyon:hardware — grid5000,," 2017. [Online]. Available: https://www.grid5000.fr/mediawiki/index.php?title=Lyon:Hardware

[13] C. Gunaratne, K. Christensen, and B. Nordman, "Managing energy consumption costs in desktop pcs and lan switches with proxying, split tcp connections, and scaling of link speed," *Int. J. Netw. Manag.*, vol. 15, no. 5, pp. 297–310, Sep. 2005.

[14] I. Cuadrado Cordero, A.-C. Orgerie, and C. Morin, "GRaNADA: A Network-Aware and Energy-Efficient PaaS Cloud Architecture," in *IEEE International Conference on Green Computing and Communications (GreenCom)*, Sydney, Australia, Dec. 2015.

[15] D. Delaney, T. Ward, and S. McLoone, "On consistency and network latency in distributed interactive applications: A survey–part i," *Presence: Teleoper. Virtual Environ.*, vol. 15, no. 2, pp. 218–234, Apr. 2006.

[16] J. Nielsen, *Usability engineering*. Elsevier, 1994, ch. 5.

[17] J. Johnson, *GUI bloopers: donts and dos for software developers and Web designers*. Morgan Kaufmann, 2000, ch. 7.

[18] F. Consulting, "Akamai reveals 2 seconds as the new threshold of acceptability for ecommerce web page response times," https://www.akamai.com/us/en/about/news/press/2009-press/akamai-reveals-2-seconds-as-the-new-threshold-of-acceptability-for-ecommerce-web-page-response-times.jsp, 2009.

[19] J. D. Brutlag, H. Hutchinson, and M. Stone, "User preference and search engine latency," *JSM Proceedings, Qualtiy and Productivity Research Section, Alexandria, VA*, 2008.